Congratulations! You passed!

Grade received 100% **To pass** 80% or higher

Go to next item

Introduction to Regular Expressions in SQL

Latest Submission Grade 100%

1. The **%** wildcard character is a substitute for _____? **1 / 1 point**

- a single character
- a sequence of characters
- a sequence of metacharacters
- all rows and columns

Correct
Correct! The **% (percentage sign)** is used as a substitute for a sequence of characters.

2. Write a query to extract all individuals from the employees table whose **lastname** does not contain the characters **'Kin'**. **1 / 1 point**

```
SELECT * FROM employees
WHERE lastname NOT LIKE ('%Kin%');
```

```
SELECT * FROM employees
WHERE last_name NOT LIKE ('%Kin%');
```

```
SELECT * FROM employees
WHERE lastname LIKE ('%Kin%');
```

```
SELECT * FROM employees
WHERE firstname NOT LIKE ('%Kin%');
```

Correct
Correct! This query will correctly extract all individuals from the employees table whose **lastname** does not contain the characters **'Kin'**.

3. Which of the following query is correct to retrieve a list of all customers whose **city** starts with **T** in POSIX regular expression **(*assuming the city name is case sensitive*)?** **1 / 1 point**

```
SELECT * FROM customers
WHERE cty ~* 'T+[a-z\s]+$';
```

```
SELECT * FROM customers
WHERE cty ~ '^T+[a-z\s]+$';
```

```
SELECT * FROM customers
WHERE city ~* '^T+[a-z]+$';
```

```
SELECT * FROM customers
WHERE city ~ '^T+[a-z\s]+$';
```

Correct
Correct! This query correctly retrieves a list of all customers whose **city** starts with **T** in POSIX regular expression **(*assuming the city name is case sensitive*)?**

4. The **SIMILAR TO** operator returns true or false depending on whether its pattern matches the given string **1 / 1 point**

- Yes
- Sometimes
- No

Correct
Correct! The SIMILAR TO operator returns true or false depending on whether its pattern matches the given string. It is similar to LIKE, except that it interprets the pattern using the SQL standard's definition of a regular expression.

5. As an SQL user with a basic idea of using regular expressions in SQL, retrieve the **firstname**, **lastname**, phone number **(phone)**, and **email** of all customers whose **email addresses contain two-digit numbers** using regular expressions. ***(Select all that apply)*** **1 / 1 point**

- [x]
```
SELECT firstname, lastname, phone, email
FROM customers
WHERE email ~ '[0-9][0-9]';
```

Correct
Correct! The query above retrieves the **firstname**, **lastname**, phone number **(phone)**, and **email** of all customers whose **email addresses contain two-digit numbers** using regular expressions.

- [ ]
```
SELECT firstname, lastname, phonenumber, email
FROM customers
WHERE email ~ '[0-9][0-9]';
```

- [x]
```
SELECT firstname, lastname, phone, email
FROM customers
WHERE email ~ '[0-9]{2}';
```

Correct
Correct! The query above retrieves the **firstname**, **lastname**, phone number **(phone)**, and **email** of all customers whose **email addresses contain two-digit numbers** using regular expressions.

- [ ]
```
SELECT firstname, lastname, phone, email
FROM customers
WHERE email ~ '{0-9}(2)';
```

6. The **SUBSTRING()** function used in the POSIX regular expression takes how many parameters? **1 / 1 point**

- one
- three
- None of the above
- two

Correct
Correct! The **SUBSTRING() function with two parameters**, **SUBSTRING(*string* FROM *pattern*)**, provides extraction of a substring that matches a POSIX regular expression pattern. It returns null if there is no match, otherwise the portion of the text that matched the pattern.

7. Which of the following is a valid parameter for the **regexp_matches()** function? ***(Select all that apply)*** **1 / 1 point**

- [x] pattern

Correct
Correct! The regexp_matches() function returns a text array of all of the captured substrings resulting from matching a POSIX regular expression pattern. It has the syntax **regexp_matches(string, pattern [, flags])**.

- [x] flags

Correct
Correct! The regexp_matches() function returns a text array of all of the captured substrings resulting from matching a POSIX regular expression pattern. It has the syntax **regexp_matches(string, pattern [, flags])**.

- [x] string

  Correct
  Correct! The regexp_matches() function returns a text array of all of the captured substrings resulting from matching a POSIX regular expression pattern. It has the syntax **regexp_matches(string, pattern [, flags])**.

- [ ] expressions